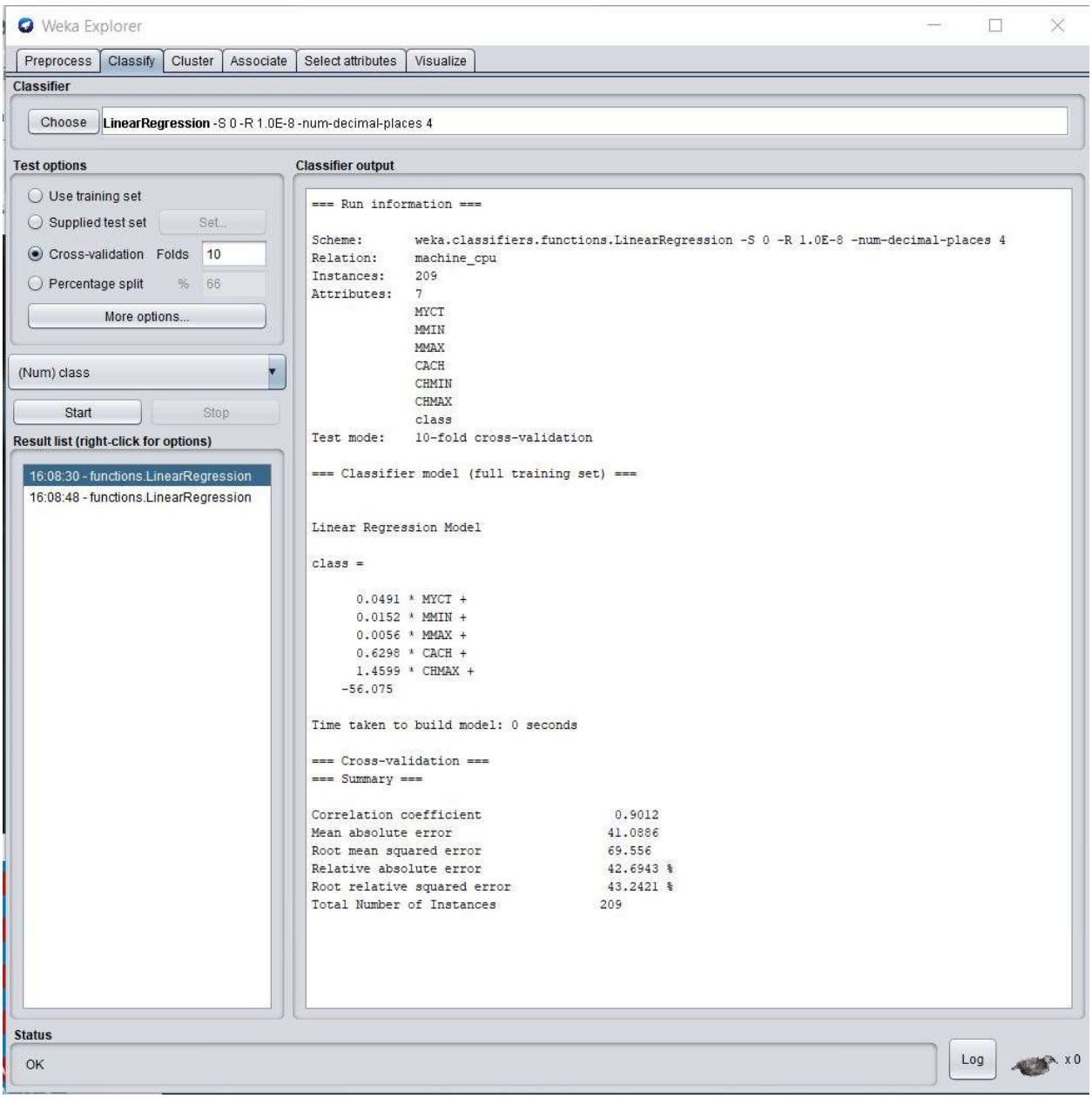
Alan Chen

ECE 331 Project 2

Machine Learning (Linear Regression), Matrix Multiplication, and Cache Simulation in MIPS assembly language

11/22/2019

## WEKA OUTPUT (for weights and bias value)



## MIPS Results vs Hand Calculated Results

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Weights | 49 | 15 | 6 | 630 | 0 | 1460 | Bais Value: | -56075 |
| Data | 125 | 256 | 6000 | 256 | 16 | 128 | | |
| Data*Weights | 6125 | 3840 | 36000 | 161280 | 0 | 186880 | Ans | 338 |
| Data | 29 | 8000 | 32000 | 32 | 8 | 32 | | |
| Data*Weights | 1421 | 120000 | 192000 | 20160 | 0 | 46720 | Ans | 324 |
| Data | 29 | 8000 | 16000 | 32 | 8 | 16 | | |
| Data*Weights | 1421 | 120000 | 96000 | 20160 | 0 | 23360 | Ans | 205 |

Data Segment

| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) | Value (+1c) |
|---|---|---|---|---|---|---|---|---|
| 0x10010900 | 64 | 12 | 16 | 26 | 16000 | 32000 | 64 | 16 |
| 0x10010920 | 24 | 26 | 16000 | 32000 | 64 | 8 | 24 | 26 |
| 0x10010940 | 8000 | 32000 | 0 | 8 | 24 | 26 | 8000 | 16000 |
| 0x10010960 | 0 | 8 | 16 | 480 | 96 | 512 | 0 | 1 |
| 0x10010980 | 1 | 203 | 1000 | 2000 | 0 | 1 | 5 | 338 |
| 0x100109a0 | 324 | 324 | 324 | 204 | 344 | 464 | 464 | 656 |
| 0x100109c0 | 983 | 0 | 3 | 77 | 155 | -31 | 104 | -1 |
| 0x100109e0 | 35 | 1 | 237 | 57 | 63 | 116 | 126 | 15 |
| 0x10010a00 | -13 | 8 | -11 | 7 | 4 | 98 | 98 | 102 |

There was one error introduced by our fixed point approximation (204 vs 205) because for hand counted result, I rounded 204.98 to 205 whereas in MIPS, I just ignore the remainder.

## MIPS(Predicted) Results vs Actual Results

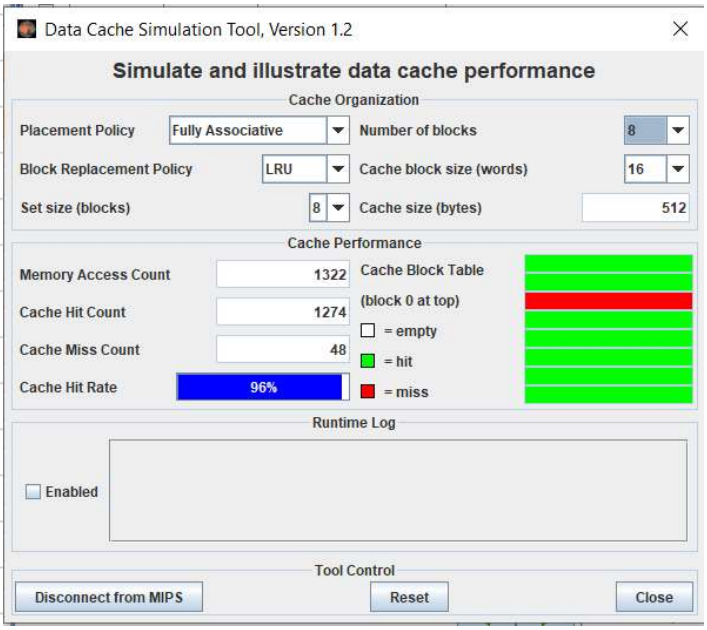
```
@data
125,256,6000,256,16,128,198
29,8000,32000,32,8,32,269
29,8000,32000,32,8,32,220
29,8000,32000,32,8,32,172
29,8000,16000,32,8,16,132
26,8000,32000,64,8,32,318
23,16000,32000,64,16,32,367
```

| Results | |
|---|---|
| Predicted | Actual |
| 338 | 198 |
| 324 | 269 |
| 205 | 132 |

The linear regression function use for predictions are not that accurate. I think this is because the points that I have chosen are far from the line of best fit, since linear regression considers all values.

# Optimizing Cache Performance Using Cache Simulator

| # of Blocks | Block Length | Level of Associativity | Hit Rate |
|---|---|---|---|
| 8 | 4 | Direct Mapping | 67% |
| 8 | 8 | Direct Mapping | 76% |
| 8 | 16 | Direct Mapping | 83% |
| 16 | 16 | Direct Mapping | 91% |
| 4 | 4 | 4-Way Set Associative | 58% |
| 4 | 8 | 4-Way Set Associative | 83% |
| 8 | 16 | 4-Way Set Associative | 96% |
| 16 | 8 | 4-Way Set Associative | 85% |
| 8 | 4 | Fully Associative | 86% |
| 8 | 8 | Fully Associative | 93% |
| 8 | 16 | Fully Associative | 96% |
| 16 | 4 | Fully Associative | 86% |




Fully Associative would have the best cache performance for level of associativity because you can put data anywhere within the cache whereas in Direct Mapping and 4-Way Set Associativity, you can only put data in certain blocks. Also the longer the block length is, the better the cache hit rate will be. This is because the you are bringing in more data from main memory which will increase the hit rate. From a bit level perspective, longer block length will increase the offset bit which will decrease the tag bits, therefore the tag bits will more likely be the same within the same block.